```
# response time per condition

setwd('/home/allgoodguys/Documents/Studying/Lund_PhD/epistles/
005_with-Lima/analysis')
library (lme4)
require(brms)
require(shinystan)

df = read.csv ('data/data_preprocessed.csv')[,-1]
df = df[df$type=='target',] # targets only
df = df[df$correct==T & df$outlier==F,] # only correct responses
analyzed
df$log_time2 = log(df$time2)

aggregate(time2~condition, df, mean)
aggregate(time2~condition, df, sd)
boxplot (time2~condition, df) # answer more slowly only in
deliberated (1300 vs ~900 ms)

a = aggregate(time2~subject,df,function(x)median(x,na.rm=T))
hist(a[,2])


mod0 = glmer(log_time2~condition+(1|item)+(1|subject), data=df,
family="gaussian")
summary (mod0)
drop1(mod0, test='Chisq')
plot(mod0)
qq = qqnorm(resid(mod0))
cor(qq$x,qq$y) # 0.99 for log_time2 / gaussian (0.95 for time2/
gaussian)

mod0_1 = glmer(log_time2 ~ condition + pretestAccuracy + (1|item)
+(1|subject), data=df, family="gaussian")
drop1(mod0_1, test='Chisq')

mod0_1 = glmer(log_time2 ~ condition*pretestAccuracy + (1|item)+(1|
subject), data=df, family="gaussian")
drop1(mod0_1, test='Chisq')

mod1 = glmer(log_time2~condition*subjIntensity +
condition*pretestAccuracy+(1|item)+(1|subject), data=df,
family="gaussian")
drop1(mod1, test='Chisq')

mod2 = glmer(log_time2 ~ condition*pretestAccuracy + subjIntensity +
(1|item)+(1|subject), data=df, family="gaussian")
drop1(mod2, test='Chisq')
anova(mod0, mod1, mod2)
summary(mod2)

### brms time2~cond
mod_overall_full = brm(time2 ~ condition*pretestAccuracy +
subjIntensity + (1|item)+(1|subject), data=df, family="lognormal",
```

```
warmup=100, iter=500, chains=4, cores=4, save_ranef=F,
prior=set_prior('normal(0,5)'))
# saveRDS(mod_overall_full, 'models/mod_time2~cond*pretest.RDS')
# mod_overall = readRDS('models/mod_time2~cond*pretest.RDS')
plot(mod_overall_full)
summary(mod_overall_full)

newdata = expand.grid(
  subjIntensity = median(df$subjIntensity),
  pretestAccuracy = seq(min(df$pretestAccuracy),
                        max(df$pretestAccuracy),
                        length.out = 100),
  condition = levels(df$condition)
)
fit = fitted(mod_overall_full, newdata = newdata, re_formula=NA)
colnames(fit) = c('fit', 'se', 'lwr', 'upr')
df_plot = cbind(newdata, fit)
df_plot$pretestAccuracy = df_plot$pretestAccuracy * 100  # to %
ggplot(df_plot, aes(x = pretestAccuracy, y = fit, ymin = lwr, ymax =
upr, color = condition, fill = condition)) +
  geom_line(size = 2) +
  geom_ribbon(color = 'white', alpha = .15) +
  ylab('Latency, ms')


coda = posterior_samples(mod_overall_full)
colnames(coda)
betas = list(
  beta_pretest_delib = coda[, 5],
  beta_pretest_fast = coda[, 5] + coda[, 7],
  beta_pretest_load1 = coda[, 5] + coda[, 8],
  beta_pretest_load2 = coda[, 5] + coda[, 9]
)

sapply(betas, function(x) quantile(x, probs = c(.5, .025, .975)))

# contrasts
beta_pretest_delib_vs_rest = quantile(
  betas$beta_pretest_delib - (
    betas$beta_pretest_fast + betas$beta_pretest_load1 +
betas$beta_pretest_load2
  ) / 3,
  probs = c(.5, .025, .975)
)
beta_pretest_delib_vs_rest

beta_pretest_load1_vs_load2 = quantile(
  betas$beta_pretest_load1 - betas$beta_pretest_load2,
  probs = c(.5, .025, .975)
)
beta_pretest_load1_vs_load2

beta_pretest_fast_vs_load = quantile(
  betas$beta_pretest_fast - (betas$beta_pretest_load1 +
```

```
betas$beta_pretest_load2) / 2,
  probs = c(.5, .025, .975)
)
beta_pretest_fast_vs_load
# etc

# w/o intensity and pretest accuracy
mod_overall = brm (log_time2~condition+(1|item)+(1|subject),
data=df, family="gaussian", warmup=100, iter=500, chains=4, cores=4,
save_ranef=F, prior=set_prior('normal(0,5)'))   #
# saveRDS(mod_overall, 'models/mod_time2~cond.RDS')
# mod_overall = readRDS('models/mod_time2~cond.RDS')
# stancode(mod)

plot(mod_overall)
summary(mod_overall)
# launch_shiny(mod_overall)

coda = posterior_samples(mod_overall)
colnames(coda)

a = data.frame(
  delib = coda[,1],
  fast = coda[,1] + coda[,2],
  load1 = coda[,1] + coda[,3],
  load2 = coda[,1] + coda[,4]
)
a = apply (a, 2, function(x)exp(x))
# write.csv (a, 'mcmc_time2~cond.csv')

df_plot_overall = data.frame (emotion=rep('Overall',4),
condition=c('Deliberated','Fast','Load 1','Load 2'), lwr=NA, fit=NA,
upr=NA)
df_plot_overall[,3:5] = t ( apply (a, 2,
function(x)quantile(x,probs=c(.025,.5,.975))) )

ggplot(df_plot_overall, aes(x=condition, y=fit))+
  geom_point(stat='identity', shape=1, size=3) +
  geom_errorbar(aes(ymin=lwr, ymax=upr)) +
  scale_fill_discrete(name='') +
  xlab('') +
  ylab('time2, ms') +
  theme_bw ()

# ggsave ('pix/time2~cond.jpg', width=7, height=4, units='cm',
dpi=300, scale=1.5)

# contrasts
colnames(a)
quantile (a[,1] - (a[,2]+a[,3]+a[,4])/3, probs=c(.025,.5,.975)) #
delib vs rest
quantile (a[,2] - (a[,3]+a[,4])/2, probs=c(.025,.5,.975)) # fast vs
load
quantile (a[,4] - a[,3], probs=c(.025,.5,.975)) # load 1 vs load 2
```

```
### time2 ~ condition*emotionBlock
### time2~emotion
aggregate(time2~emotionBlock, df, mean)
boxplot (time2~emotionBlock, df)

mod_te = glmer(log_time2~emotionBlock+(1|item)+(1|subject), data=df,
family="gaussian")
summary(mod_te)
drop1(mod_te, test='Chisq') # yes

aggregate (time2~condition+emotionBlock, df, mean)
mod0 = glmer(log_time2~condition*emotionBlock+(1|item)+(1|subject),
data=df, family="gaussian")
summary (mod0)
drop1 (mod0, test='Chisq') # yes, with interaction
plot(mod0)
qq = qqnorm(resid(mod0))
cor(qq$x,qq$y) # 0.99


mod = brm (log_time2 ~ condition*emotionBlock + (1|subject)+(1|
item), data = df, warmup=100, iter=1000, chains=4, cores=4,
family='gaussian', ranef=F, prior=set_prior("horseshoe(1)"), control
= list(adapt_delta = 0.99)) # set_prior for all beta-coef at once
(shrinkage): see https://cran.r-project.org/web/packages/brms/
vignettes/brms.pdf
# saveRDS(mod, 'models/mod_time2~cond*em_shrinkage.RDS')
# mod = readRDS('models/mod_time2~cond*em_shrinkage.RDS')
# stancode(mod)

plot(mod)
summary(mod)
# launch_shiny(mod)

coda = posterior_samples(mod)
colnames(coda)

a = data.frame(achievement_delib=coda[,1],
               achievement_fast=coda[,1]+coda[,2],
               achievement_load1=coda[,1]+coda[,3],
               achievement_load2=coda[,1]+coda[,4],

               amusement_delib=coda[,1]+coda[,5],
               amusement_fast=coda[,1]+coda[,2]+coda[,5]+coda[,12],
               amusement_load1=coda[,1]+coda[,3]+coda[,5]+coda[,13],
               amusement_load2=coda[,1]+coda[,4]+coda[,5]+coda[,14],

               anger_delib=coda[,1]+coda[,6],
```

```
                anger_fast=coda[,1]+coda[,2]+coda[,6]+coda[,15],
                anger_load1=coda[,1]+coda[,3]+coda[,6]+coda[,16],
                anger_load2=coda[,1]+coda[,4]+coda[,6]+coda[,17],

                disgust_delib=coda[,1]+coda[,7],
                disgust_fast=coda[,1]+coda[,2]+coda[,7]+coda[,18],
                disgust_load1=coda[,1]+coda[,3]+coda[,7]+coda[,19],
                disgust_load2=coda[,1]+coda[,4]+coda[,7]+coda[,20],

                fear_delib=coda[,1]+coda[,8],
                fear_fast=coda[,1]+coda[,2]+coda[,8]+coda[,21],
                fear_load1=coda[,1]+coda[,3]+coda[,8]+coda[,22],
                fear_load2=coda[,1]+coda[,4]+coda[,8]+coda[,23],

                pleasure_delib=coda[,1]+coda[,9],
                pleasure_fast=coda[,1]+coda[,2]+coda[,9]+coda[,24],
                pleasure_load1=coda[,1]+coda[,3]+coda[,9]+coda[,25],
                pleasure_load2=coda[,1]+coda[,4]+coda[,9]+coda[,26],

                relief_delib=coda[,1]+coda[,10],
                relief_fast=coda[,1]+coda[,2]+coda[,10]+coda[,27],
                relief_load1=coda[,1]+coda[,3]+coda[,10]+coda[,28],
                relief_load2=coda[,1]+coda[,4]+coda[,10]+coda[,29],

                sadness_delib=coda[,1]+coda[,11],
                sadness_fast=coda[,1]+coda[,2]+coda[,11]+coda[,30],
                sadness_load1=coda[,1]+coda[,3]+coda[,11]+coda[,31],
                sadness_load2=coda[,1]+coda[,4]+coda[,11]+coda[,32])
a = apply (a, 2, function(x)exp(x))

df_plot = data.frame (emotion =
rep(c('Achievement','Amusement','Anger','Disgust','Fear','Pleasure',
'Relief','Sadness'),each=4),
condition=rep(c('Deliberated','Fast','Load 1','Load 2'),8), lwr=NA,
fit=NA, upr=NA)
df_plot[,3:5] = t ( apply (a, 2, function(x)quantile(x,probs=c(.
025,.5,.975))) )
df_plot = rbind (df_plot, df_plot_overall)

# re-arrange the order of emotions:
df_plot$emotion = factor(df_plot$emotion,
levels=c('Achievement','Amusement','Pleasure','Relief','Anger','Disg
ust','Fear','Sadness','Overall'))

# re-label conditions
df_plot$condition = as.character(df_plot$condition)
df_plot$condition[df_plot$condition=='Load 1'] = 'Low load'
df_plot$condition[df_plot$condition=='Load 2'] = 'High load'
df_plot$condition = factor(df_plot$condition,
levels=c('Deliberated','Fast','Low load','High load'))

ggplot (df_plot, aes(x=emotion, y=fit, fill=condition)) +
  geom_bar(stat='identity', position=position_dodge(.7), width=.7) +
  geom_errorbar(aes(ymin=lwr, ymax=upr), position=position_dodge(.
```

```
7), width=.5) +
  geom_vline(xintercept=4.5, linetype=2) +
  geom_vline(xintercept=8.5, linetype=2) +

scale_fill_manual(name='',values=c('gray70','gray50','gray30','gray1
0')) +
  xlab("") +
  ylab('Latencies for Hits (ms)') +
  theme_bw () +
  theme(legend.position="top", panel.grid=element_blank())

# ggsave ('pix/time~cond*em.jpg', width=10, height=5, units='cm',
dpi=300, scale=2)


# sanity check:
df_plot_ctrl = aggregate (time2~condition+emotion, df, median)
ggplot (df_plot_ctrl, aes(x=emotion, y=time2, fill=condition)) +
  geom_bar(stat='identity', position=position_dodge(.7)) +
  scale_fill_discrete(name='') +
  xlab("Actual data") +
  ylab('time2, ms') +
  theme_bw () +
  theme(legend.position="top")


## contrasts
mcmc0 = read.csv('mcmc_time2~cond.csv')[,-1]
# Is "deliberated" overall better than the other three conditions?
colnames(a)
delib_vs_rest = data.frame (ems =
c('Achievement','Amusement','Anger','Disgust','Fear','Pleasure','Rel
ief','Sadness'), fit=NA, lwr=NA, upr=NA, stringsAsFactors=F)
for (em in 1:nrow(delib_vs_rest)){
  r = 4*em-3
  delib_vs_rest[em,2:4] = quantile( a[,r]-(a[,r+1]+a[,r+2]+a[,r+3])/
3, probs=c(.5,.025,.975))
}
delib_vs_rest = rbind (delib_vs_rest,
c('Overall',quantile( mcmc0[,'delib']-(mcmc0[,'load1']
+mcmc0[,'load2']+mcmc0[,'fast'])/3, probs=c(.5,.025,.975))))

# Is load6 better than load8?
load6_vs_8 = data.frame (ems =
c('Achievement','Amusement','Anger','Disgust','Fear','Pleasure','Rel
ief','Sadness'), fit=NA, lwr=NA, upr=NA, stringsAsFactors=F)
for (em in 1:nrow(load6_vs_8)){
  r = 4*em
  load6_vs_8[em,2:4] = quantile( a[,r-1]-a[,r], probs=c(.
5,.025,.975))
}
load6_vs_8 = rbind (load6_vs_8,
c('Overall',quantile( mcmc0[,'load1']-mcmc0[,'load2'], probs=c(.
5,.025,.975))))
```

```
# Is fast better than mean(load6,load8)?
fast_vs_load = data.frame (ems =
c('Achievement','Amusement','Anger','Disgust','Fear','Pleasure','Rel
ief','Sadness'), fit=NA, lwr=NA, upr=NA, stringsAsFactors=F)
for (em in 1:nrow(fast_vs_load)){
  r = 4*em-2
  fast_vs_load[em,2:4] = quantile( a[,r]-(a[,r+1]+a[,r+2])/2,
probs=c(.5,.025,.975))
}
fast_vs_load = rbind (fast_vs_load,
c('Overall',quantile( mcmc0[,'fast']-(mcmc0[,'load1']
+mcmc0[,'load2'])/2, probs=c(.5,.025,.975))))


### Valence, arousal, intensity, pretest accuracy
mod_intens = glmer(log_time2~condition*subjIntensity + duration +
(1|item)+(1|subject), data=df, family="gaussian")
summary(mod_intens)
drop1(mod_intens, test='Chisq')  # marginal interaction

mod_intens1 = glmer(log_time2~condition+subjIntensity+duration + (1|
item)+(1|subject), data=df, family="gaussian")
summary(mod_intens1)
drop1(mod_intens1, test='Chisq')

mod_pretest = glmer(log_time2~condition*pretestAccuracy + (1|item)
+(1|subject), data=df, family="gaussian")
summary(mod_pretest)
drop1(mod_pretest, test='Chisq')

mod_pretest1 = glmer(log_time2~condition*pretestAccuracy +
subjIntensity + (1|item)+(1|subject), data=df, family="gaussian")
summary(mod_pretest1)
drop1(mod_pretest1, test='Chisq')

mod_pretest2 = glmer(log_time2~condition*pretestAccuracy +
subjIntensity + subjValence + subjArousal + (1|item)+(1|subject),
data=df, family="gaussian")
summary(mod_pretest2)
drop1(mod_pretest2, test='Chisq')  # drop arousal

mod_pretest3 = glmer(log_time2~condition*pretestAccuracy +
subjIntensity + subjValence + (1|item)+(1|subject), data=df,
family="gaussian")
summary(mod_pretest3)
drop1(mod_pretest3, test='Chisq')

anova(mod_pretest, mod_pretest1, mod_pretest2, mod_pretest3)  #
mod3, definitely



### Non-Bayesian significance testing to verify
```

```
# signif per condition
library(nlme)
m1 = lme(log_time2~condition, random=list(item=~1, subject=~1),
data=df)
summary(m1)

# fast vs. load1+load2
df1 = df[df$condition != 'Deliberated', ]
df1$fast = df1$condition == 'Fast'
mod = lme(log_time2~fast, random=list(item=~1, subject=~1),
data=df1)
summary(mod)
anova(mod)

# low vs. high load
df$condition = relevel(df$condition, ref = 'Load 1')
m1 = lme(log_time2~condition, random= list(item=~1, subject=~1),
data=df)
summary(m1)

# duration
mod_dur = glmer(log_time2~condition*duration + (1|item)+(1|subject),
data=df, family="gaussian")
summary(mod_dur)
drop1(mod_dur, test='Chisq')
```